REMARKS

The Examiner is thanked for his careful and very thorough Office Action. The Examiner is particularly thanked for the helpful suggestions regarding correction of the alleged informalities.

Claims 1-7 have been rejected. By the foregoing amendments, various Claims are sought to be amended or canceled without prejudice.

Claims 8-20 have been added. The support for Claims 8, 9, 14, 17 and 20 can be found, for example, in the paragraph beginning on page 11, line 34 and Figure 3. The support for Claims 10-13, 16 and 19 can be found, for example, in the paragraph beginning on page 73, line 2 and Figure 2. The support for Claim 15 can be found, for example, in the paragraph beginning on page 8, line 31. The support for Claim 18 can be found, for example, in the paragraph beginning on page 8, line 11. The new claims are respectfully asserted not to introduce new matter, and their entry is respectfully requested.

Art Rejections

The art rejections are all respectfully traversed.

Review of the References

Some of the major technical differences between the references applied and the disclosure of the present application will now be reviewed. Of course, these points in the specification do not define the scope or interpretation of any of the claims; they are listed merely to help appreciate the importance of the claim distinctions that will be reviewed thereafter.

Peddada et al.

Peddada et al. (U.S. Patent No. 6,295,068) relates to a 3D graphics driver that manages a texture cache in the local graphics memory. The driver disclosed by *Peddada et al.* does not appear to manage texture memory in the main memory nor does it suggest managing page faulting of texture data. *Peddada et al.* also does not disclose allowing the graphics accelerator itself

direct access to the main memory or any other innovations with regard to graphics accelerators.

Porterfield

Porterfield (U.S. Patent No. 6,249,853) relates to a modular device for storing, addressing, and retrieving graphics data from main memory. The graphics accelerator chip disclosed by *Porterfield* does not contain a memory management function.

If the undersigned attorney has overlooked a relevant teaching in any of the references, the Examiner is requested to point out very specifically where such teaching may be found.

Rejection Under 35 USC 103(a)

Claims 1-7 stand rejected under 35 USC Section 103(a) as being unpatentable over *Peddada et al.* in view of *Porterfield.*

Claim 6 has been canceled without prejudice by the above amendment to expedite prosecution. The rejection of this claim is traversed and is now believed to be moot.

The asserted combination of references does not support each limitation of Claim 1. Specifically, Claim 1 recites "**a graphics accelerator unit which manages page faulting of texture data invisibly to the host processor**".

Although *Peddada et al.* relates to graphics processing, it does not appear to teach or suggest any innovations with regard to graphics accelerators. *Peddada et al.*, instead, relates a 3D graphics driver that manages a texture cache in the local graphics memory. As correctly noted by the Examiner, *Peddada et al.* fails to explicitly suggest or teach managing page faulting of texture data invisibly to the host processor.

The Examiner has suggested that it would have been obvious to combine the AGP model of *Peddada et al.* with the AGP Execute model of *Porterfield* in order to reduce the size, and thus the cost, of the local frame

buffer and to improve system performance as taught by *Porterfield*. However, it is not understood how combining the two AGP models would result in the advantage suggested by the Examiner. *Peddada et al.* even expressly teaches away from the suggested combination:

> *Unfortunately, 3D graphics accelerator 20 must have additional hardware to directly access textures from AGP memory 14. This extra hardware adds to the expense and complexity of 3D graphics accelerator 20 and is thus undesirable.*[1]

A prior art reference may be considered to teach away when:

> [A] person of ordinary skill, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken by the applicant.[2]

Peddada et al. only mentions the AGP Execute model in the background of the invention in order to show how it improves upon the prior art of the AGP Execute model. Therefore, the Examiner's suggestion that it would have been obvious to combine the two AGP models is not only unfounded but is also contradictory to the teachings of *Peddada et al.* Accordingly, the Examiner has failed to establish a proper motivation for the suggested combination.

Neither of the applied references discloses or suggests managing page faulting of texture data invisibly to the host processor. The Examiner has suggested that GART is similar in function to paging hardware in the CPU chip. However, the page fault management disclosed in the present

---

[1] Col. 1., *ll.* 58-62.

[2] *In re Gurley*, 27 F.3d 551, 553, 31 USPQ 2d 1130, 1131 (Fed. Cir. 1994).

application goes beyond the GART used by AGP protocols. As stated in the present application:

> Intel's built-in chip set hardware is called the GART (Graphics Address Remapping Table). The GART hardware is somewhat similar in function to the paging hardware in the CPU chip, in that the processor "linear" virtual addresses get automatically translated into physical addresses (which may point to system RAM and local Frame Buffer memory, as well as the AGP RAM).
>
> However, this translation is fairly inflexible, and completely out of the user's control. Thus it cannot be optimized for particular applications, software architectures, or graphics accelerator architectures[3]...
>
> [T]he present application discloses a computer system in which a graphics accelerator unit manages page faulting of texture data invisibly to the host processor. When a logical page fault occurs and the page of texture is in the second level of memory (i.e. the host's physical memory) it will be fetched in automatically by the graphics memory manager, and the host is not aware anything has happened.[4]

---

[3] Page 4, line 28 – Page 5, line 2.

[4] Page 8, *ll.* 11-15.

Neither of the applied references, or any motivated combination thereof, discloses such graphics accelerator. Therefore, even if one were somehow motivated to combine the two AGP models (which Applicant strongly disputes), the suggested combination still would not support all the limitations of Claim 1.

As determined in *Thrift*,[5] a rejection which "does not discuss the unique limitations" of the claims was held to be "simply inadequate on its face." In this case, a rejection was held "not supported by substantial evidence because the cited references do not support each limitation of claim 11." See *In re Vaeck*, 947 F.2d 488, 493, 20 USPQ2d 1438, 1443 (Fed.Cir. 1991)."[6] Therefore, a prima facie case of obviousness has not been established by the Examiner with regard to Claim 1.

Claim 2 also recites features not shown or suggested by the asserted combination. Specifically, Claim 2 recites "**a graphics accelerator unit which manages page faulting of texture data, from dedicated graphics memory into a main memory used by at least one host processor, invisibly to the host processor, except when said graphics accelerator unit calls for data which has not recently been present in said main memory**".

As stated above, neither of the applied references, or any motivated combination thereof, discloses or suggests a graphics accelerator that is capable of managing page faulting of texture data, from a dedicated graphics memory into a main memory used by a host processor, invisibly to the host processor. Therefore, a prima facie case of obviousness has not been established by the Examiner with regard to Claim 2.

Claim 3 also recites features not shown or suggested by the asserted combination. Specifically, Claim 3 recites "**a graphics accelerator unit, comprising rendering accelerator logic,**

---

[5] *In re Thrift*, 298 F.3d 1357 (Fed.Cir. 2002).

[6] *In re Thrift*, 298 F.3d at 1366 (emphasis added).

dedicated graphics memory, and a second memory management unit which manages texture data for said accelerator logic and performs page faulting of said texture data, invisibly to said CPU."

Again, neither of the applied references, or any motivated combination thereof, discloses or suggests a graphics accelerator that is capable of performing page faulting of texture data invisibly to the host processor. Therefore, a prima facie case of obviousness has not been established by the Examiner with regard to Claim 3.

Claim 4 also recites features not shown or suggested by the asserted combination. Specifically, Claim 4 recites "a graphics accelerator unit having respective local memory associated therewith, and also having a graphics memory manager; when said graphics accelerator unit attempts to access texture data which is in said physical memory associated with said host, said graphics memory manager fetches said texture data automatically".

Neither of the applied references, or any motivated combination thereof, discloses or suggests a graphics accelerator having a graphics memory manager that automatically fetches texture data from the host's physical memory. Therefore, a prima facie case of obviousness has not been established by the Examiner with regard to Claim 4.

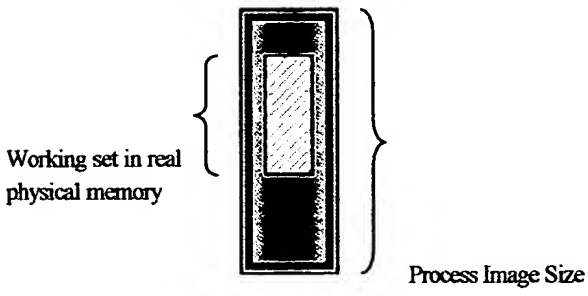
Claim 7 also recites features not shown or suggested by the asserted combination. Specifically, Claim 7 recites "a host processor having host physical memory associated therewith, and also having virtual memory management; and a graphics accelerator unit having respective physical memory associated therewith, and also having virtual memory management; and wherein, when said graphics accelerator unit attempts to access texture data which is in said host physical memory, if said texture data is in said host physical memory, said

graphics memory manager fetches said texture data therefrom automatically; and if said texture data is not in said host physical memory, said texture data is first loaded into said host physical memory, and thereafter said graphics memory manager fetches said texture data automatically from said host physical memory".

Memory management can be viewed as a collection of techniques for providing sufficient memory to one or more processes in a computer system, especially when the system does not have enough memory to satisfy all processes' requirements simultaneously. Techniques include swapping, paging, and virtual memory.[7]

Virtual memory, as employed by the present application, presents a level of indirection between the addresses that an application views, and the physical memory addresses used by the hardware. The benefits of virtual memory include: security, reliability, application transparent relocation of physical memory, and cache partitioning.

Virtual Memory Management loads all process images in parts called "Working-Sets".[8]



[7] This particular teaching is illustrated at http://www.hyperdictionary.com/dictionary/memory+management.

[8] This particular teaching illustrated at http://iris.nyit.edu/~spatanga/Mem_Mangt_swap.doc.

Virtual memory uses disk storage space to make the computer work as if it had more memory. When a file or program is too big for the computer to work with in its memory, part of the data is stored on disk. This virtual storage is divided into segments called pages; each page is correlated with a location in physical memory, or RAM. When an address is referenced, the page is swapped into memory; it is sent back to disk when other pages must be called. This allows the program to run as if all the data is in memory.[9]

Neither of the applied references, or any motivated combination thereof, discloses or suggests a host having virtual memory management or a graphics accelerator unit having virtual memory management with a graphics memory manager capable of automatically fetching texture data from the host's physical memory. Therefore, a prima facie case of obviousness has not been established by the Examiner with regard to Claim 7.

Finally, dependent Claim 5, which depends directly from independent Claim 4 and incorporates all the limitations thereof, also includes additional limitations that are not shown or suggested by the asserted combination.

Specifically, Claim 5 recites "**wherein, after fetching said texture data, said graphics memory manager restarts texture processing**".

Thus, for the reasons discussed above, Applicant respectfully requests withdrawal of this rejection.

---

[9] This particular teaching is illustrated at http://www.computeruser.com/resources/dictionary/definition.html?lookup=5891.

Conclusion

Thus, all grounds of rejection and/or objection are traversed or accommodated, and favorable reconsideration and allowance are respectfully requested. The Examiner is requested to telephone the undersigned attorney or Robert Groover for an interview to resolve any remaining issues.

Respectfully submitted,

N. Elizabeth Pham, Reg.No. 49,042

Customer Number 29106

Attorney for Applicant

11330 Valley Dale Drive, Dallas TX 75230

214-363-3038 *groover@technopatents.com* March 2, 2004